Automatic hexahedral-dominant meshing for decomposed geometries of complex components

Automatic Hexahedral-Dominant Meshing for Decomposed Geometries of Complex Components

Benoit Lecallard [0000-0003-0904-5443][1], Christopher M. Tierney [0000-0003-3341-6902][2], Trevor T. Robinson [0000-0002-6595-6308][3], Cecil G. Armstrong [0000-0001-8695-5016][4],Liang Sun [0000-0002-7900-4447][5], Declan C. Nolan [0000-0002-9388-6183][6] and Alexander E. Sansom[7].

[1]Queen's University Belfast, blecallard01@qub.ac.uk
[2] Queen's University Belfast, christopher.tierney@qub.ac.uk
[3] Queen's University Belfast, t.robinson@qub.ac.uk
[4] Queen's University Belfast, c.armstrong@qub.ac.uk
[5] Queen's University Belfast, liang.sun@qub.ac.uk
[6] Queen's University Belfast, d.nolan@qub.ac.uk
[7]Rolls-Royce Plc, Alexander.Sansom@rolls-royce.com

Corresponding author: Trevor T. Robinson, t.robinson@qub.ac.uk

ABSTRACT

An equivalent non-manifold cellular model is used to enrich manifold decompositions of a CAD model to create a model suitable for finite element analysis. Thin-sheet and long-slender decomposition tools are integrated around the common data structure in order to automatically define a meshing recipe based on analysis attributes identified during the decomposition. Virtual topology operations are used to replicate the hard geometry splits in the non-manifold representation and create a robust bi-directional mapping between manifold and non-manifold representations. Adjacency information extracted from the non-manifold cellular model, alongside the appropriate analysis attributes and linear integer programming methods, are used to define a hex-dominant meshing recipe, which can then be applied to automatically generate a mesh.

Keywords: Database, Non-manifold, Hexahedral-dominant meshing.

1 INTRODUCTION

Generating good quality simulation models is a major bottleneck in the automation of simulation workflows. It can often be the most time consuming task in the design process and can require extensive user effort and skills. As a result, the use of simulation tools throughout the analysis cycle is not as prevalent as it could be, in particular at early stages of the design process, where the configuration is prone to modifications and the cost of updating the simulation model is prohibitive. Having clear information on the simulation objectives and the ability to automate analysis model setup based on

these is the key to streamline the downstream analysis process. Simulation intent, defined by Nolan et al. [11], aims to capture all the analysis, modelling and abstraction decisions in order to derive an analysis model from an initial CAD geometry. This is achieved using three main technologies: Cellular Modelling [1] to assign analysis attributes to each cell in a non-manifold sub-division of the model; Equivalencing to maintain associativity between models at different level of abstraction; and Virtual Topology [15] for creating fit-for-analysis models defined at a topological level without affecting the original geometry. However, tools are required to manage the analysis attributes attached to cells in the cellular model and to automatically generate the analysis models at the desired levels of abstraction and detail, with the proper couplings and constraints applied.

Different element types are often preferred depending on the physics to be solved or the shape properties of the geometry to be analysed. Many analysis workflows require the use of hexahedral (hex) elements in order to accurately analyse the highly non-linear time-dependent events such as crash or gas turbine fan-blade off. At the same time, hex elements can handle anisotropy better than tetrahedral (tet) elements, hence reducing the number of degrees of freedom (DOFs) of the model in highly anisotropic regions. While tet mesh generators are already very robust and highly automated, automatic hex mesh generation producing good quality elements still requires significant user effort for complex components. In the past decade, promising methods such as using frame-fields [5, 6, 7] have been developed for hex meshing of general geometry. But it still requires more fundamental understanding of the singularities in order to achieve a valid hex mesh. The current industry standard for generating hex meshes consists in manually sub-dividing the design geometry into sweep-meshable sub-domains. Several automatic decomposition tools have been developed to address this issue [2], [8], [22], [17] and [18]. Many automated decomposition tools don't fully decompose the solid body into hex-meshable blocks, and therefore a mesh is obtained either by manually decomposing the leftover regions [8], or creating a hex-dominant mesh by automatically tet-meshing the regions to which a hex mesh cannot be applied.

Even though the decomposition of certain geometries can be automated, the meshing is still not straightforward. One reason is that splitting the geometry using standard geometric operations in a CAD-type solid modelers results in loss of information, as the manifold structure of CAD environments cannot retain the interfaces between cells. Even though interfaces could be recovered using Boolean operations, it is a computationally expensive operation which is highly sensitive to tolerances and can result in the creation of sliver entities. While dedicated CAE tools such as Hypermesh or ICEM do not have this issue, they fail to retain the construction tree of the original CAD model since it has been designed in another package before being imported. Generating the meshing recipe at the CAD level enables to keep advanced geometric manipulation capabilities and to keep access to functional information and parameters contained in the feature tree. Besides, as no all-hex meshing tool is available yet, the problem of how to interface meshes between different size and/or types of elements arises. Non-conformal interfaces, where there is no exact match between the nodes on either sides can be handled by formulating multi-point constraints (MPC) equations to couple the DOFs between the nodes. However, this method is computationally expensive and the solution loses accuracy, which is a problem because the interface regions are often critical zones for stress. Conforming interfaces between

tet and hex mesh is achievable by inserting pyramid elements, after the common interface has been properly specified, to ensure all the nodes are merged. These transition elements can either be created by node insertion or using the pyramid open method [12].

Finally, the sizing of the mesh remains a mostly manual task, and while much research has been done on mesh adaptation and automatic sizing, it doesn't totally address the problem of propagating size variation through decomposed models. White and Tautges proposed a toolkit to automatically identify meshing strategies [21], while Tam and Armstrong used integer programming to ensure mesh compatibility for collection of connected sub-regions or primitives [19]. Understanding how the number of edge subdivisions propagate through the model can also help identify independent zones in the mesh, and the reduced number of constraints can be used to divide the meshing task between teams or tools, making the mesh generation process faster [21]. The interval assignment problem can be solved using integer programming, where the optimum number of elements is identified for the set of constraints.

The objective of this work is to automate the decomposition and meshing steps of an analysis workflow. The simulation intent for this workflow is to generate an all-solid hex-dominant mesh where so-called thin-sheet and long-slender regions receive a structured mesh by sweeping and residual regions are tet-meshed. Starting from the design CAD model, a fully automated approach is built on top of the automated decomposition approaches described in [17],[18].

To enhance the process herein, the manifold decomposition is enriched by generating an equivalent non-manifold cellular topological representation. This representation uses virtual topology operations to track the subdivision history thus capturing information lost through the manifold decomposition. In addition, each cell in the non-manifold cellular decomposition is assigned appropriate analysis attributes related to the geometric reasoning tool used to dictate the decomposition. The enriched common data structure (CDS) uses integer programming routines and adjacency information from the cellular model to automatically create a hex-dominant mesh with correct mesh controls and mesh-mating.

2 PROCESS OVERVIEW

Proper management of analysis information such as interfaces and meshing strategy is essential to successfully automate analysis workflows, especially when it involves many different software tools. In particular, the different representations used in CAD and CAE packages make the mapping of entities challenging. CAD systems often use a manifold representation while many CAE systems use a non-manifold or polygonal representation. The key difference for this work is that in a manifold representation, a face can only bound one body. As a result, the entities at the interface between two bodies are not readily available and must be identified using a series of geometric queries. In a non-manifold structure, a single face can be shared by two bodies at their interface, enabling it to be retrieved using simple topological queries. Exploiting this fact and creating a single mesh definition at the interface can ensure a conformal mesh is generated between adjacent bodies.

The solution presented herein is an independent topological definition of the CAD and CAE representation, as depicted in Fig. 1. CAD and CAE representations are linked to one another through a

common data structure which enables analysis attributes to be transferred between them. Geometry manipulations are carried out only on appropriate boundary topology within the data structure, facilitating the creation of a virtual non-manifold analysis representation. Storing geometry modifications in this manner links the manifold and non-manifold models without requiring the use of expensive Boolean operations.

The enriched data structure is then used to automatically derive the meshing recipe necessary to generate a hex-dominant mesh from the manifold decomposition.

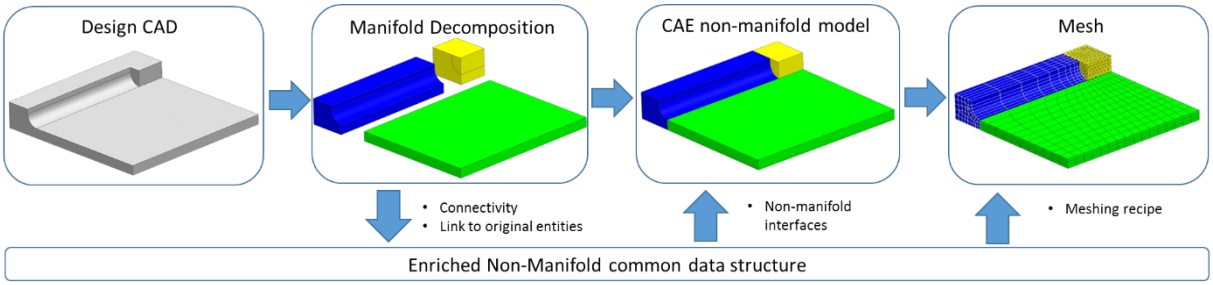

Figure 1: Automatic hex-dominant meshing process using an external common data structure (CDS) relating CAD manifold and CAE non-manifold representations.

3 CAPTURING DECOMPOSITION DECISIONS

3.1 Thin-Sheet and long-slender decomposition

The fully automated method for hex-dominant mesh generation from the design CAD model presented here is built on top of the automated decomposition approaches described in [17], [18]. These describe a two-step process to identify and isolate different classes of sweep-able regions in a CAD model.

First, thin-sheet regions (those which have two dimensions larger than the third) are extracted by interrogating and manipulating the pairs of large opposing faces bounding the candidate region (Fig. 2(b)). The face-pairs are projected one onto another, and their intersection is calculated in the parametric space of the largest one. The aspect ratio is checked, and entities in close proximity are merged to avoid creating sliver entities. Then, appropriate cutting surfaces are defined to isolate the thin regions. Once all the cutting surfaces are defined, the geometry is partitioned using split operations in the CAD environment. The next step is to designate entity attributes to facilitate the downstream sweeping operation. Source and target (S/T) faces are subsets of original face pairs, and all faces connecting them are defined as wall faces. In the geometry in Fig. 2(a), two thin regions can be extracted.

In a second decomposition step, long-slender regions (those which have one dimension larger than the other two, blue in Fig. 2(g)) are extracted. They are identified from the residual regions of the thin-sheet decomposition by searching for loops of nearly parallel long edges. First, edges are classified as long by comparing their length with the lateral dimension of the bounded faces. Then loops of long edges are identified, which in turn are converted into loops of wall faces. Next, the necessary cutting surfaces are defined, with an offset to avoid creating poor quality geometry in the residual regions. Finally, source

and target faces are identified to facilitate sweep meshing. As before, all the splits are done once all the cutting surfaces have been defined.

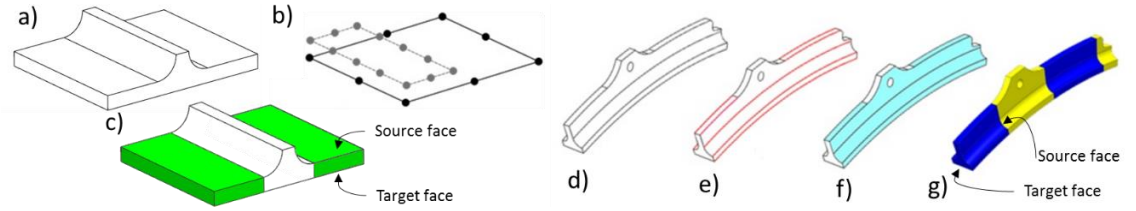

Figure 2: Thin-sheet [17] (left) and long-slender [18] (right) decomposition: (a) candidate geometry, (b) discretized face pair, (c) two thin-sheet regions extracted in green, (d) candidate geometry, (e) long-edges identification, (f) 2 loops of long faces, and (g) two long-slender regions extracted in blue.

While this decomposition can significantly reduce the manual effort required to generate the mesh, it is not appropriate for a good quality automatic meshing process. The first issue is that the interface information is not retained since the manifold representation used in most CAD packages will generate a set of disconnected bodies upon decomposition. If the mesh is manually generated, the non-manifold capabilities of CAE environments enable all the bodies to be reconnected after the user manually specifies the interfaces. In our automated workflow, interface entities must be tracked in order to generate the meshing recipe without any user intervention, hence a mapping between manifold and non-manifold interfaces is recorded in the cellular model.

Moreover, dissimilar pairs of faces can appear at the interfaces, as imprints are not propagated during successive decompositions. This is something which is not considered in previous research. In Fig. 3(b), the split operation generates a pair of manifold faces at the interface between the two new bodies. Since this is a manifold representation, the two subsequent split operations in Fig. 3(c) each divide only one of the faces at the interface, hence a dissimilar interface exists between adjacent bodies. For example, the red face at the non-manifold interface in Fig. 3 (c) is required to obtain a conformal mesh at the interface between bodies but is missing from the CAD model. However, a topological face entity is created in the non-manifold cellular model from the non-manifold edges which can be easily retrieved from the manifold edges, allowing the correct meshing recipe to be identified. Secondly, it is difficult to record adjacency information from the decomposition process, since the decomposition is non-binary. A split can result in more than two bodies, and all the splits are done at the end of the identification process. As a result, manual intervention is required to ensure mesh conformity at the interfaces.

Finally, the decomposition process returns all the sweepable bodies, and what their source and target faces are. At this stage, mesh generation still requires significant user intervention to obtain a good quality mesh, and to avoid incompatibilities between the source and target faces for sweeping the elements.

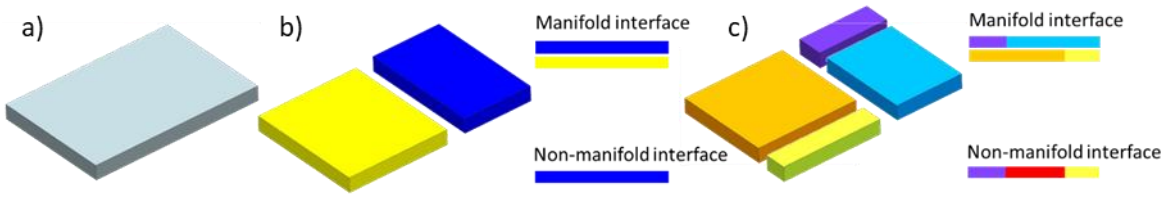

Figure 3: (a) starting block, (b) blocks and interface after one split, either of the manifold interface can represent the non-manifold interface, and (c) after another set of splits, a non-conformal interface appears. The non-manifold interface cannot be simply identified from the manifold interface and the face in red is missing.

3.2 Enriched common data structure

The issues presented earlier are applicable to most decompositions tools. Since a variety of them can be used sequentially to construct an automated meshing workflow, it is important to robustly capture all the geometric modifications and to keep tracks of the interfaces.

The proposed approach makes use of an external common data structure (CDS) stored in a SQL relational database, as proposed by Tierney et al. [20], to link different representations and store analysis attributes attached to cells. A non-manifold cellular representation of the model is created in the database by applying the virtual topology operations corresponding to the decomposition of the original topology based on [17] and [18]. Therefore, adjacency information for the decomposed volumes is automatically retained and missing interfaces are captured by the non-manifold nature of the cellular model.

Since cutting surfaces in a hard split re-use the model topology whenever possible, most of the non-manifold entities exist in the manifold CAD representation. As a result, the non-manifold representation in this work is obtained by editing the entities (vertices/edges/faces) and their bounding and bounded entities in the CDS when available, and missing faces are recovered by looking at open loops of subset edges. This can be achieved since all the non-manifold edges can be easily recovered from the set of manifold edges. Once in the non-manifold CAE environment, all the interfaces are automatically specified from the CDS and a one-to-one correspondence is obtained between the topological representation contained in the CDS and the topology of the model in the CAE package. It is necessary to keep track of the entities specific to the manifold representation that remain in the CAD package (for example the orange and light blue faces in Fig. 3(c)) to maintain a bi-directional link between the manifold and non-manifold representation.

A virtual topology relation in the database also records the history of the decomposition by mapping the analysis topology to the original manifold design topology, hence linking the decomposed model with the design model. For example, partitioned edges, faces and volumes are stored as subsets of their original host entities. Different identifiers are required to robustly map entities across packages, especially since the model can be converted to different geometry types (e.g. polygon faces and edges) in CAE packages. Vertices are referred to by their coordinates and edges by their midpoints. Higher dimension entities are identified by queries on their bounding entities topology and orientation.

3.3 Capturing geometry modification

This section describes how the non-manifold cellular model contained in the CDS is generated using virtual topology operations. First, relationships between entities before and after split operation need to be identified, then topology manipulations are carried out virtually in the CDS, using the propagate topology algorithm to duplicate the split operation.

During the decomposition, all the cutting surfaces are generated first, then all the splits are performed. One solution to track topological changes is to identify the interaction between the cutting surface topology and the CAD model to be decomposed. This can be achieved using geometric queries, such as point containment methods. A vertex from the cutting tool lying on an edge of the target is used to split this edge, and the same applies to edges lying on faces and faces inside volumes, provided they extend until the boundaries. However, these interrogations are expensive to compute, and the complexity of the search algorithms is non-linear. This information could also be extracted from the decomposition tools when the cutting surface is generated, however it requires case-by-case modification of the tools and doesn't offer a generic approach for the automated workflow. In this work partitioned entities are tracked using the call-back function in the geometric kernel (Parasolid [13] in this work). User-specified call-back functions are defined to automatically identify whether a split operation has occurred and return the entities that have been partitioned and the splitting entities. This link between a partitioned entity and its original entity is stored in the database within the virtual topology relation.

Once all of the entities affected by the decomposition have been identified, they are classified into four categories (see Fig. 4 for examples). The first three categories are identical to the one defined in [15] for virtual topology, and the fourth one is a combination of the first two which is added to have a better control on a specific configuration.

- Parasite entities: entities that did not exist in the original topology but lie on an existing entity of higher dimension (i.e. an edge laying on the face it splits). These entities are created at the manifold interface, and therefore need to be matched and paired to characterize an interface.
- Split entities: subsets of host entities that are split by a parasite entity
- Orphan entities: entity without host (i.e. an edge bounding only parasite faces)
- Partially existing entities: parasite topology entity which is also a subset of an entity (i.e. a bounding edge of a parasite face which is a subset of an existing edge). This category can only contain edges and corresponds to the case where a subset of an entity from the original model is reused to define the cutting tool topology. In this case the entity is treated as a split entity then as a parasite.

Once all the entities affected by the decomposition have been identified, virtual topology manipulations are applied to duplicate the decomposition on the non-manifold cellular model, using the following algorithm:

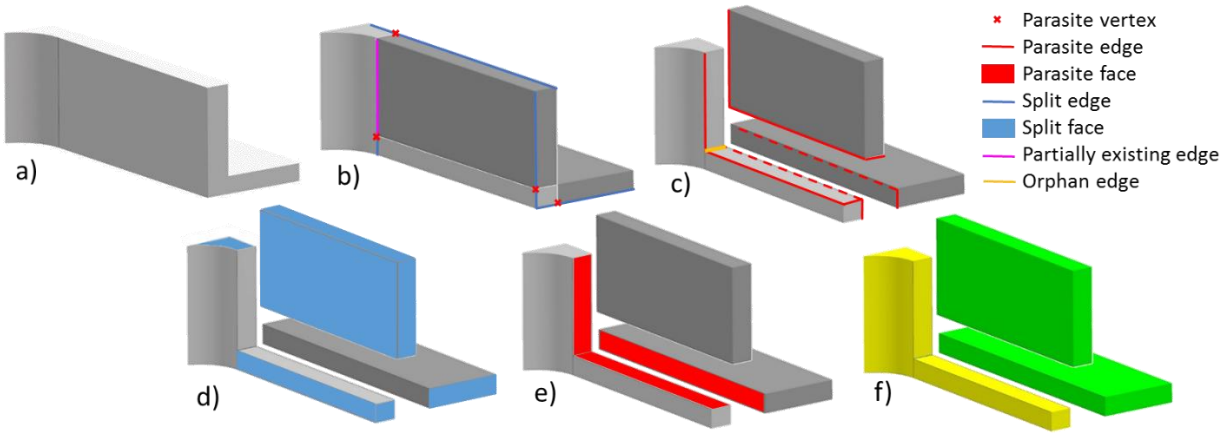

Figure 4: Propagation of a thin-sheet decomposition

Propagate topology algorithm:

- Sort entities into those modified or created by the split operation
- Propagate split edges (Fig. 4(b))
  - Insert parasite vertex and split edges in the database
  - Update faces bounded by the edge
  - Remove host entity from analysis topology
- Propagate parasite edges (Fig. 4(c))
  - Match pair of manifold edges
- Record unmatched edges (dashed red lines in Fig. 4(c))
  - Virtual split to create non-manifold entities
  - Keep the smallest subset
- Propagate split faces (Fig. 4(d))
  - Insert new subset faces into database, all edges are already existing
  - Write bounding entities topological relationships using the minimal subsets
- Match adjacent parasite faces (Fig. 4(e))
  - Match pairs of manifold faces and generate non-manifold in database
  - Write bounding topology of parasite face
- Record unmatched faces (to handle configurations similar to in Fig.3(c))
  - Stored in order to link manifold and non-manifold representations
- Propagate split bodies (Fig. 4(f))
  - Write bounding topology and store in virtual topology as subsets of host entities

After the CDS has been enriched by the decomposition process, it can be used to inform downstream processes to generate a meshing recipe and create the mesh.

3.4 Analysis Attributes

Besides the definition of the partitioning strategy, decomposition tools also provide analysis attributes attached to the cells created by the split, which are used to enrich the non-manifold cellular representation. These analysis attributes, such as the definition of thin-sheet, long-slender and residual

regions, help define appropriate meshing strategies and are shown in Tab.1. Analysis variables are also attached to the attributes to store important parameters such as the aspect ratio of the identified region. Thin-sheets are sweep-meshable, and the important parameter to drive the sizing of the mesh in the sweep direction is the number of elements through the thickness. Long-slender regions inherit their sizing constraints from the neighbouring thin-sheets. Thin-sheet and long-slender regions have a source and a target face for the sweep, and the source face can be either mapped or paved with quad elements. In order to fully constrain the hex element dimensions, analysis attributes need to be attached to those faces. Mapping or paving can be left to the discretion of the automatic mesh recipe generator to facilitate the simulation intent and generate a fit-for-purpose mesh. For this work, residual regions often don't have any simple hex-meshing strategy attached and are tet-meshed.

Once all analysis attributes have been identified, they are stored in the cellular model. This reduces downstream reasoning by utilizing the information from the decomposition tool and also helps transfer the identified meshing strategy to the mesh generation process. Other parameters, such as the aspect-ratio of thin-sheets calculated during the decomposition, are utilised to aid mesh size identification.

Table 1: Analysis attributes extracted during the decomposition.

Analysis attribute	*Mesh type*	*Method*	*Analysis variable*
Thin-sheet (TS)	Hex	Swept	Aspect ratio, number in thickness
Long-slender (LS)	Hex	Swept	Sweep direction sizing inherited from TS
Residual (R)	Tet	Automatic Tet	Sizing inherited from adjacent TS and LS
Source faces	Quad	Mapped/paved	Aspect ratio

4 MESHING RECIPE

4.1 Connectivity Graph and Configurations

In order to ensure a good quality mesh, the mesh metrics have to account for the geometry configuration and properties, as well as the connectivity between the different cells. For example, an edge shared by two thin-sheets which is in the sweep direction of one thin region and bounds a source face of the other thin region (source-wall edge in Fig.5(c) and (a)) identify an area where a denser mesh is likely to be required. This suggests a transition region might be necessary in one of the thin-sheet regions in order to provide a smooth transition in mesh density. The topology contained in the CDS contains all the interface information, since a non-manifold representation is stored. Simple topological adjacency queries, such as the common boundary between entities of a specific dimension, are used to identify the connectivity graph of the different cells, as shown in Fig. 5 (b). Using the attributes identified by the

decomposition tool, Tab. 2 and 3 are defined to determine what configuration corresponds to the connectivity pattern and which analysis attribute (Tab. 4) should be applied to initialize the meshing recipe.

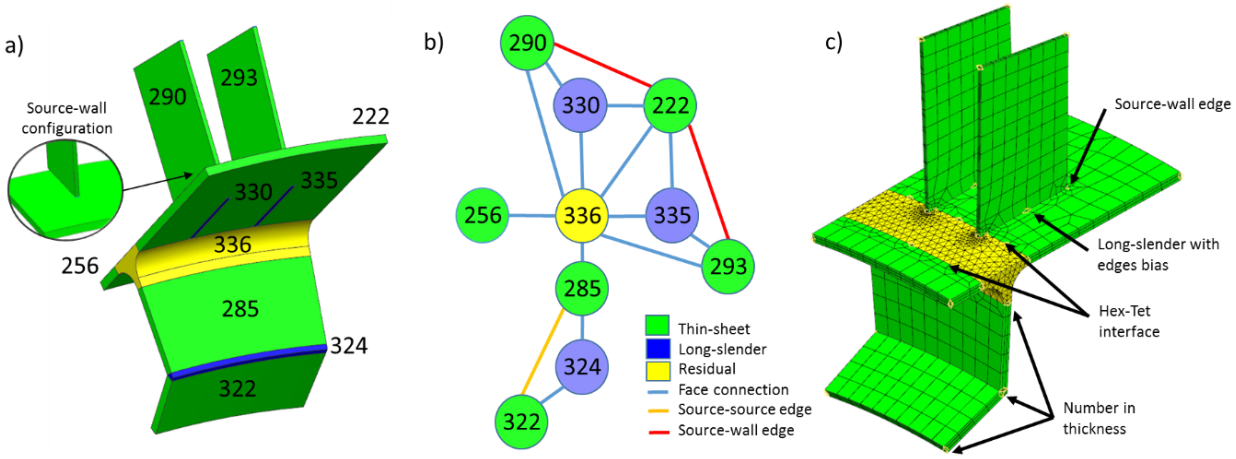

Figure 5: (a) decomposed model, (b) connectivity graph, and (c) configurations identified.

The meshing attributes are applied for specific connectivity patterns. However, the number of possible configurations can be very large and therefore the meshing strategy attached to cells is exploited to define the meshing order and recipe. Therefore, the problem can be reduced based on the decomposition rules, since some configurations will never occur given the decomposition strategy used (void cells in Tab. 2 and 3). For example, two residual regions can't share a face in our decomposition process, since the result will only ever be a single complex region. In a non-manifold representation volume cells can be connected by vertices, edges or by faces. In this work, compatible hex meshes are generated without needing to assign meshing attributes at volume interfaces that only consist of vertices. However, such situations are easily identified within the non-manifold database and could be incorporated to define boundary conditions, such as point loads. Analysis attributes are assigned to edges and faces as described below.

When cells share an edge, the situation can be much more complex, as the edges can bound any number of bodies with different analysis attributes, and therefore with different edge attributes. For example, using the decomposition shown in Fig. 5(a), edges can bound both thin-sheet and/or long-slender regions, in which case they can either be in the sweep direction (wall edge) or bound a source or target face (S/T edge). In this example the edges can bound up to three bodies with three different types. Edges which are not at an interface between bodies are classified as free. On the connectivity graph in Fig. 5(b), a face connection link implies that all the edges bounding the face are also connecting the two bodies, but these connections are not displayed for information. Tab. 2 shows all the possible configurations for edges linking 2 bodies only, and the corresponding analysis attributes, given in Tab.4.

Table 2: Analysis attributes from edge connectivity (see Tab. 4 for details).

Edge Connectivity		Long-Slender		Thin-Sheet		Residual	Free
		S/T	Wall	S/T	Wall		
Long-Slender	S/T	G	-	G	HS	G	G
	Wall	-	L*	L	HS	-	L*
Thin-Sheet	S/T	G	L	L	HS + T	L+T	L
	Wall	HS	HS	HS+T	HS	HS	HS
Residual		G	-	L+T	HS	-	G
Free		G	L*	L	HS	G	-

A face can bound a maximum of two bodies, therefore the number of configurations is smaller than for the edges and an example related to the decomposition presented above is shown in Tab. 3. For n types of cells, there are $\frac{1}{2}n(n+1)$ configurations possible, since the interface is symmetric. In this work, there are 5 types of face cells (long-slender source/target, long-slender wall, thin-sheet source/target, thin-sheet wall and residual) derived from the body cell types, plus a free type for faces that are not interfaces, resulting in 21 possible configurations. This number is reduced to 11 by removing all the configurations which do not comply with the decomposition rules and therefore will not arise.

Table 3: Analysis attributes from face connectivity (see Tab. 4 for details).

Face Connectivity		Long-Slender		Thin-Sheet		Residual	Free
		S/T	Wall	S/T	Wall		
Long-slender	S/T	SO	-	-	M+B	Py	M/P
	Wall	-	-	-	M	-	M
Thin-sheet	S/T	-	-	-	-	-	M/P
	Wall	M+B	M	-	M	Py	M
Residual		Py	-	-	Py	-	-
Free		M/P	M	M/P	M	-	-

All the analysis attributes shown in Tab. 4 stem from the analysis attributes identified during the decomposition (Tab. 1). These attributes need to comply with the meshing strategy assigned to the different cells, for example all wall faces need to be mapped meshed to comply with sweeping constraints. Another objective is to use the anisotropic properties of different regions to stretch the elements so that they are suited to the region they model, hence reducing the number of DOFs. For example, elements on the wall edges of a long-slender region can be grown along the length of the region. Analysis variables such as the number of divisions, or which quad mesher to use, are initialised as described in section 4.2.2 and attached to the meshing attributes, which will be optimised in a later step to generate the final meshing recipe.

Table 4: Analysis attributes identified from cellular model interrogation.

Analysis attribute	*Mesh control*	*Analysis variable*
B	Bias on Edge	Growth ratio to limit aspect ratio of elements near the connected ends of a long slender
G	Edge density	Global size based on smallest local size L
HS	Edge density	Hard set number through thickness of thin-sheet
L/L*	Edge density	Local size based on nearby thin-sheet (*if no thin-sheet connected, long-slender use their own aspect ratio)
M	Quad mesher	Mapped mesh
T	Allowable growth ratio of elements	Transition zone required (potential offset)
P	Quad mesher	Paver
Py	Tet mesher	Pyramid transition elements
SO	-	Sweeping order for chains of sweepable bodies

However, the symmetry of the interface is not a valid assumption in the case where multiple sweep-able bodies are connected by their source and target faces. In the example of Fig. 6, a sweep-able body with a sub-mappable wall face is decomposed into a sequence of simpler sweep-able bodies, which need to share the same sweeping direction to avoid incompatible meshes. This issue is addressed by traversing and identifying chains of sweep-able bodies to define a different set of attributes to store the sequence for successive meshing.

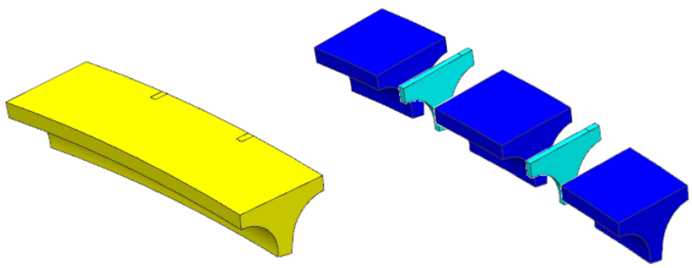

Figure 6: Sequence of sweep-able bodies, only one source face must be quad meshed.

The connectivity graph also enables interfaces between tet regions and hex regions to be processed by inserting pyramid transition elements ensure a conformal mesh at the interface. Poor quality elements generated by the change of element size between the isotropic and the anisotropic regions are avoided by defining a transition zone. Fig. 7 shows an example of different aspect ratios of pyramid elements which are used at the hex-tet transition.

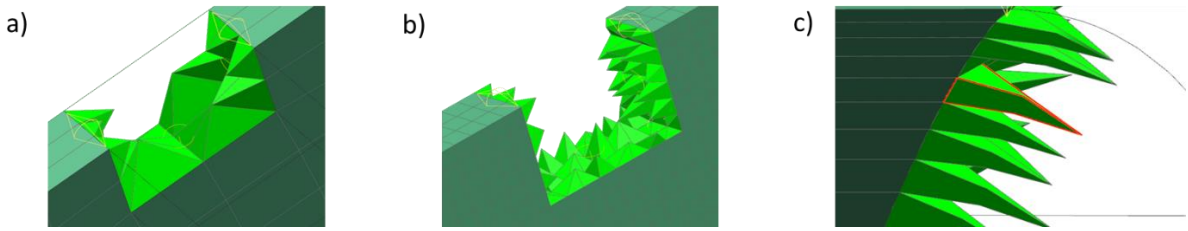

Figure 7: Pyramid transition elements: (a) aspect ratio =5, (b) aspect ratio =1, (c) failed element.

4.2 Interval Assignment Problem

4.2.1 Problem formulation

The quad meshes used for sweep meshing impose certain constraints on the number of elements or intervals of the bounding edges of a face. Each wall face needs to be mapped meshed to comply with the sweeping constraint, while quad meshes on source faces are obtained by either mapping or paving.

The interval assignment problem is formulated into a linear integer program and solved [10] in order to define suitable and compatible element division numbers on edges. This is achieved in four steps:

i) The problem is initialized by finding the number of variables and which of them should be optimised;

ii) Each edge is assigned an initial number of divisions, or goal, based upon the geometric properties of the owning body;

iii) The constraints which control how the sizing propagates throughout the model are extracted from the interface information in the database and processed according to the configurations listed in Tab. 2 and Tab. 3;

iv) The problem is solved and the results are filtered to remove unrealistic constraints before adding them to the meshing recipe stored in the database. The aim of the optimization is to identify a set of intervals as close as possible to the targets, while ensuring all divisions throughout the model are compatible.

4.2.2 Initialisation and goals

In classic meshing approaches, a user often defines a global size and manually applies mesh controls in order to obtain a suitable mesh. Edges are assigned either a goal (soft-set) or a required (hard-set) number of intervals. Other sizing parameters are left to the discretion of the software, but these often prove to be too coarse at first, and require manual refinement. In the proposed approach, all mesh metrics are automatically derived from the attributes identified at previous stage (Tab. 1 and 4) contained in the CDS and general mesh requirements on aspect ratio or from best practice (i.e. three linear elements through thin wall to accurately model the stress distribution). In the linear program, a variable is defined for each edge, but only the edges with goals applied are optimised. Edges without goals ensure the different constraints propagate properly through the model, hence the curve will inherit the number of divisions through the constraint (for example, bounding edges of the target face

of a thin-sheet will have the division number the same as the corresponding edges on the source face). As a result, the meshing recipe fully constrains the mesh, ensuring compatibility and order independence, as well as repeatability of the mesh.

In the process presented here, aimed primarily at thin-walled structures, thin-sheet bodies drive the meshing recipe. Required numbers of divisions (HS) are assigned to wall edges of thin-sheet regions to ensure three linear elements are used through the thickness. Hard-sets are defined as an equality constraint on edges. Other goals are defined by local metrics. The division numbers of source edges of thin-sheets are defined by a target aspect ratio for the element and the aspect ratio of the region, in order to avoid over-stretched elements or overly large elements. Size metrics identified for the source faces of the thin-sheets are propagated to the bounding edges, and converted into goals (L) by querying edge lengths. For edges connecting two adjacent source faces, the densest goal is kept. Long-slender element sizes are obtained from adjacent thin-sheets when available or based on their aspect ratio otherwise (L*). Any other dimension (G), mostly on residual bodies, is defined by the size of the smallest feature of the model in order to avoid creating unnecessary small elements that would affect the time step of a transient analysis. Edges bounding element type or large size transitions (T) can either be offset or receive a modified goal, in order to limit the aspect ratio of the pyramid transition elements or poor quality hex elements.

4.2.3 Constraint identification

In this step, the constraints associated with the analysis attribute identified in Section 4.1 are translated into constraints on the number of divisions on curves in the integer programming problem, in order to ensure conformity and good mesh structure. Mapping constraints require pairs of opposite edges in a logically 4-sided face to have the same number of divisions and are straightforward to define. Sub-mapping constraints, for quad meshes of more-than-4-sided faces are more complex, but simple queries on the decomposition history and analysis attributes stored in the database can identify the most important constraints. For example, for any parasite face which needs a mapped mesh, sets of opposite edges are identified since they also bound the source or target face. Similarly, edges sharing the same host entity can be grouped in sets of opposite edges. More structured mapped mesh constraints can be identified on the source faces in order to improve the overall structure of the mesh, but this can create issues since small element size will propagate easily through the model because of the mesh structure. In order to avoid this any mapping or sub-mapping constraints with overly different goals are replaced with paving constraints. Explicit transition zones could also be defined (see section 4.2.5), and refinement templates [14] could be used to achieve a better mesh quality by limiting the number of irregular nodes.

Paving algorithms for quad mesh generation impose that for each loop of edges, the sum of the intervals is even. This constraint requires the introduction of an extra variable for each of the loop of paved faces. While it is a more flexible constraint than mapping, paving constraints can sometimes greatly affect the convergence time of the integer programming problem, when many paved faces are connected to each other.

4.2.4 *Objective function*

At this stage, the CDS contains a meshing recipe with initial guesses, and all the constraints between intervals have been identified. Hence, the linear problem can be defined, and the LPsolve solver [9] is used to optimize the intervals. In order to ensure compatibility of the meshes, any solution that meets all the constraints is sufficient to generate a mesh, however the quality could be very poor. As a consequence, it is important to choose an optimal solution by defining an objective function, which will dictate how the variables need to be adjusted. In our case, the difference to the goal is minimised after weights have been applied to encourage denser meshes [19].

The objective function is defined by Eqn. (1) and (2) as follow:

- Minimize the difference Δ_i of the variable x_i to a pre-set goal G_i

$$|x_i - G_i| = \Delta_i \qquad (1)$$

- Linearize the constraints

$$|x_i - G_i| = D_i + d_i \qquad (2)$$

  - Variables $D_i \geq 0$ and $d_i \geq 0$ are the positive respectively negative difference to the goal
  - With $D_i \geq x_i - G_i$ and $d_i \geq -x_i + G_i$

- Apply weights w_i and W_i to d_i respectively D_i (weights values are taken from [10])
  - $w_i = \frac{1.2}{G_i - 1}$ and $W_i = \frac{1}{G_i}$
  - Objective function : Minimise $\sum W_i D_i + w_i d_i$

4.2.5 *Solution and offset*

Once a solution has been identified, the edge intervals are updated in the CDS and the meshing recipe is exported to the CAE package. Wherever the difference between the targeted number of divisions (goal) and the solution given by the integer program is too large, an explicit transition zone is inserted (Fig. 8). An offset is made into the thin-sheet regions with the newly created boundary assigned the division number of the goal. The original boundary keeps the value identified by the integer program solution. The elements used to vary the size are contained in the transition zones and the rest of the body receives a more structured hex-mesh. This approach however needs considerable topological modifications in the CDS and the offset tool used in the CAD model can create robustness issues depending on the nature of the offset. Therefore, apart from simple cases, like that in Fig. 8, where offsetting operations are restricted to mainly orthogonal boundaries this explicit definition of transition zones remains open for future research.

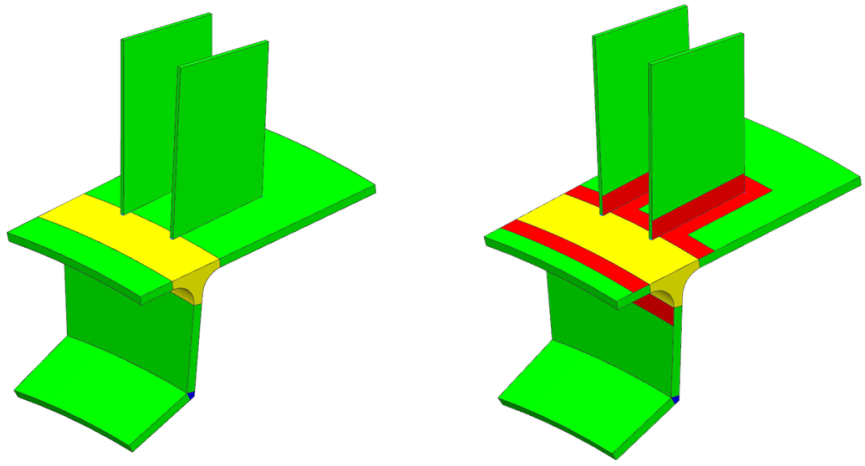

Figure 8: Decomposition (left) and corresponding transition zones by offset in red (right).

4.3 Meshing

Once the meshing recipe has been defined the model is meshed in the CAE environment. Mesh mating conditions are extracted from non-manifold adjacencies in the CDS and applied to the polygonal CAE representation within NX. After this step, there is a one-to-one correspondence between the non-manifold representation in the CAE and the representation contained in the CDS. This enables the transfer of the meshing recipe and application of the different mesh controls. This link could also be used for exchanging parametric perturbations or simulation results between the analysis model and the design model, and is a topic for future research.

Long-slender regions are meshed first since they are the most constrained, Fig. 9(b) and 9(c), followed by thin-sheets, Fig. 9(d) and 9(e). In both cases, a quad mesh is applied on a source face, and swept to the target face to generate hex elements. Then, residual regions are automatically tet-meshed, Fig. 9(f), and a layer of pyramid elements is inserted to ensure a fully conformal mesh at the interfaces with the hex-regions.

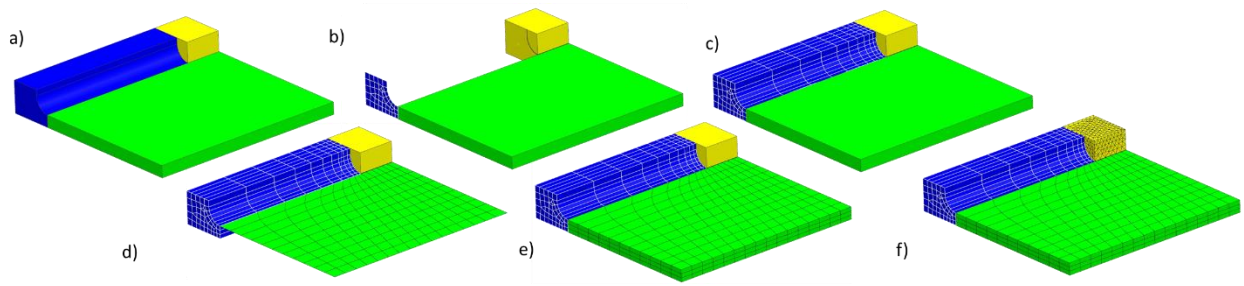

Figure 9: Meshing sequence for a simple component.

5 RESULTS AND DISCUSSION

The component in Fig. 10(a) was decomposed into 61 bodies Fig. 10(b), within 35 seconds (on a windows workstation with a 3.7 GHz Intel Xeon E5-1630 CPU with 32GB RAM). A 75% hex-dominant mesh Fig. 10(c) is obtained in 67 seconds, generating 57,000 elements. For reference, it took 4 hours to manually set-up and mesh the same decomposition, because of the difficulty to identify the correct cutting surfaces and to ensure the mesh is fully conformal at the interfaces. Fig. 10(d) shows the histogram of the Jacobian determinant, and 91% of the elements have a value above 0.6, which indicates reasonable quality is achieved for this mesh.

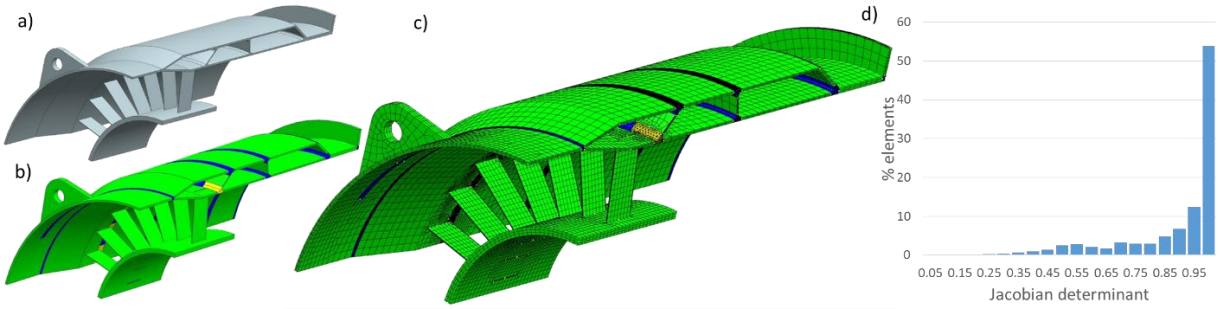

Figure 10: Automatic decomposition and meshing of a compressor casing mock-up.

Different meshes can be generated for the same component, depending on the Simulation Intent. Fig. 11(a) shows the mesh created for the thin-sheet and long-slender decomposition, Fig. 11(b) corresponds to the mesh for the same decomposition with the explicit offset from Fig. 8 applied. As mentioned previously, more work is required to properly define the explicit offset region, especially to control the element growth ratio between the regions of different mesh density. The use of the transition zones in this example allow the number of hex elements to be reduced by 30%. However, the tet elements in the residual regions still account for more than 70% of the elements. Fig. 11(c) is an all hex mesh obtained from a decomposition (Fig. 6) using successive runs of the thin-sheet and long-slender decomposition tools with different parameter variations. The total runtime was 15 seconds. This mesh has 32% less elements than the classic mixed-mesh and gives a better quality mesh. This could be further improved by using transition zones.

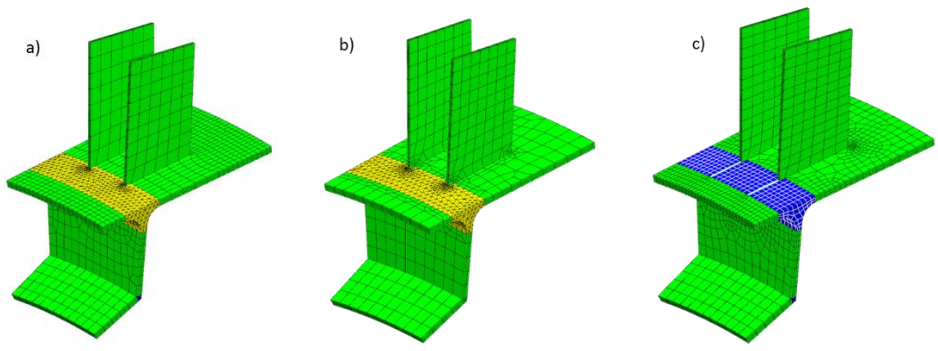

Figure 11: (a) classic mesh, (b) mesh with transition zones, and (c) all hex mesh.

Fig. 12 shows a representative aero-casing geometry which is decomposed and meshed. The model is decomposed into 1000 bodies in 21 minutes by the decomposition tools. 362,000 solid elements are created within 28 minutes by the automatic meshing tool. 52% of the elements in the mesh are hex, while less than 1% of the volume (corresponding to the 220 residual bodies) is tet-meshed. Further decomposition on some parts of the geometries suggest that a 95% hex-dominant mesh can be achieved. There are 6% of wedge elements (swept triangles) due to the fact that the paver used to generate the quad mesh can fail to generate an all-quad mesh of acceptable quality, this requiring triangles to be inserted. This also reduces the speed of the overall process. The mesh quality could be significantly improved by using multi-block decomposition on the source faces, for example using cross-fields methods [3].

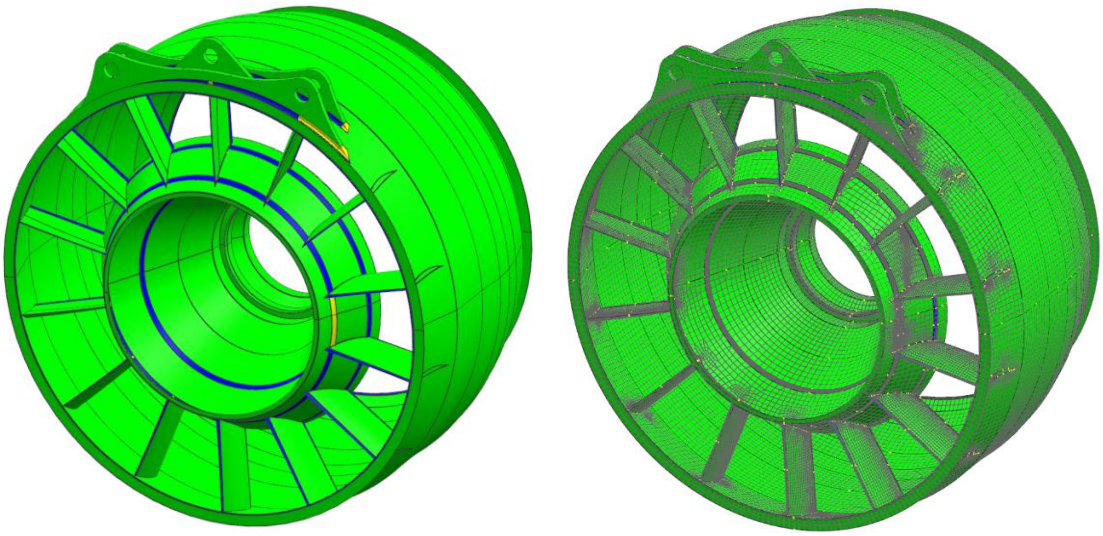

Figure 12: Decomposition and mesh of a turbine casing.

Even though the mesh generated can be used for analysis, it can still be improved further by changing the objective function in the interval assignment problem. Similar regions between the vanes on the internal hub in Fig. 12(b) receive different meshes, since minimizing the sum of the weighted deltas in Eqn. (2) tends to modify the smallest number of curves instead of aiming for the smallest modification on each curve. As result, properties such as rotational-symmetry between components are not transferred to the final mesh, however these properties could be identified at the decomposition stage, and inform the meshing recipe to reduce the complexity of the interval assignment problem and achieve a more consistent mesh. A better objective function based on minimizing the lexicographic vector of weighted differences as proposed by Mitchell [10] could potentially be used to improve the mesh quality. Finally, in this work weights are only based on the goal and therefore an overly dense mesh can be generated when a thin-sheet with a small aspect ratio connects to another one with a much larger aspect ratio. In Fig. 12(b) the outer casing could receive a coarser mesh by including a better aspect-ratio consideration when setting the goal, and the weights in the interval assignment problem.

In order to ensure a mesh is always obtained from the decomposed geometry, incompatible meshing constraints are relaxed with simpler ones, for example by replacing a mapped mesh by a paved mesh constraint on a source face, ensuring the linear program will have a feasible solution. As a last resort, triangular respectively tetrahedral elements can be used to replace failed quad respectively hexahedral regions. As a result, a mesh can always be generated, however some elements can be below the requirements of the solver in terms of quality. Improving mesh quality will be a topic of future research, as to date the focus has been to provide an automated meshing strategy and analysis workflow for the thin-sheet and long-slender decompositions.

6 Conclusions

An independent non-manifold data structure is used to manage various analysis representations. This allows a manifold decomposition of a CAD model, created for the purpose of meshing, to be enriched using virtual topology operations to record the subdivision history and also maintain robust links with the design component. A method to store and process analysis attributes is enabled to integrate different decomposition tools around a non-manifold cellular model. This allows interface information and analysis attributes to be used to automatically define the meshing recipes required to generate a hex-dominant mesh.

Acknowledgments

The authors wish to acknowledge the financial support provided by Innovate UK via GEMinIDS (project 113088), a UK Centre for Aerodynamics project. The authors acknowledge Rolls-Royce for granting permission to publish this paper. The use-case is a Rolls-Royce model from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 234344 (www.crescendo-fp7.eu).

REFERENCES

[1] Bidarra, R.; de Kraker, K. J.; Bronsvoort, W. F.: Representation and management of feature information in a cellular model, Computer-Aided Design, 30(4), 1998, 301–313. https://doi.org/10.1016/S0010-4485(97)00070-5

[2] Chong, C. S.; Senthil Kumar, A.; Lee, K. H.: Automatic solid decomposition and reduction for non-manifold geometric model generation, Computer-Aided Design, 36(13), 2004, 1357–1369. https://doi.org/10.1016/j.cad.2004.02.005

[3] Fogg, H. J.; Armstrong, C. G.; Robinson, T. T.: Enhanced medial-axis-based block-structured meshing in 2-D, Computer-Aided Design, 72, 2016, 87-101. https://doi.org/10.1016/j.cad.2015.07.001

[4] Frey, P.; George, P.: Mesh generation: application to finite elements, John Wiley & Sons, Hoboken, NJ 2008. https://doi.org/10.1002/9780470611166

[5] Huang, J.; Tong, Y.; Wei, H.; Bao, H.: Boundary aligned smooth 3D cross-frame field, Proceedings of the 2011 SIGGRAPH Asia Conference, 30(6), 2011, 143. https://doi.org/10.1145/2024156.2024177

[6] Kowalski, N.; Ledoux, F.; Frey, P.: Smoothness driven frame field generation for hexahedral meshing, Computer-Aided Design, 72, 2016, 65-77. https://doi.org/10.1016/j.cad.2015.06.009

[7] Li, Y.; Liu, Y.; Xu, W.; Wang, W.; Guo, B.: All-hex meshing using singularity-restricted field, ACM Transactions on Graphics, 31(6), 2012, 177. https://doi.org/10.1145/2366145.2366196

[8] Lu, Y.; Gadh, R.; Tautges, T. J.: Feature based hex meshing methodology: feature recognition and volume decomposition, Computer-Aided Design, 33(3), 2001, 221–232. https://doi.org/10.1016/S0010-4485(00)00122-6

[9] LpSolve, http://lpsolve.sourceforge.net/5.5/, Free Software Foundation.

[10] Mitchell, S. A.: High Fidelity interval Assignment, International Journal of Computational Geometry & Applications, 10(4), 2000, 399–415. https://doi.org/10.1142/S0218195900000231

[11] Nolan, D. C.; Tierney, C. M.; Armstrong, C. G.; Robinson, T. T.: Defining Simulation Intent, Computer-Aided Design, 59, 2015, 50–63. https://doi.org/10.1016/j.cad.2014.08.030

[12] Owen, S.; Saigal, S.: Formation of pyramid elements for hexahedra to tetrahedra transitions, Computer Methods in Applied Mechanics and Engineering, 190(34), 2001, 4505-4518. https://doi.org/10.1016/S0045-7825(00)00330-3

[13] Parasolid, https://www.plm.automation.siemens.com/en_us/products/open/parasolid/. Siemens plm software.

[14] Schneiders, R.: Refining quadrilateral and hexahedral element meshes. Transition, 2, 1996, 1.

[15] Sheffer, A.; Bercovier, M.; Blacker, T.; Clements, J.: Virtual Topology Operators for Meshing, International Journal of Computational Geometry & Applications, 10(3), 2000, 309–331. https://doi.org/10.1142/S0218195900000188

[16] Shepherd, J.; Benzley, S.; Mitchell, S.: Interval Assignment for Volumes with Holes. International Journal for Numerical Methods in Engineering, 49(1-2), 2000, 277-288. https://doi.org/10.1002/1097-0207(20000910/20)49:1/2<277::AID-NME933>3.0.CO;2-V

[17] Sun, L.; Tierney, C. M.; Armstrong, C. G.; Robinson, T. T.: Decomposing complex thin-walled CAD models for hexahedral-dominant meshing, Computer-Aided Design, (in press), 2017. https://doi.org/10.1016/j.cad.2017.11.004

[18] Sun, L.; Tierney, C. M.; Armstrong, C. G.; Robinson, T. T.: An enhanced approach to automatic decomposition of thin-walled components for hexahedral-dominant meshing, Engineering with Computers, 2017, 1-17. https://doi.org/10.1007/s00366-017-0550-x

[19] Tam T. K. H.; Armstrong, C. G.: Finite element mesh control by integer programming, International Journal for Numerical Methods in Engineering, 36(15), 1993, 2581–2605. https://doi.org/10.1002/nme.1620361506

[20] Tierney, C. M.; Nolan, D. C.; Robinson, T. T.; Armstrong, C. G.: Managing Equivalent Representations of Design and Analysis Models, Computer-Aided Design and Applications, 11(2), 2014, 193–205. https://doi.org/10.1080/16864360.2014.846091

[21] White, D. R.; Tautges, T. J.: Automatic scheme selection for toolkit hex meshing, International Journal for Numerical Methods in Engineering, 49, 2000, 127–144. https://doi.org/10.1002/1097-0207(20000910/20)49:1/2<127::AID-NME926>3.0.CO;2-V

[22] Wu, H.; Gao, S.: Automatic swept volume decomposition based on sweep directions extraction for hexahedral meshing, Procedia Engineering, 82, 2014, 136-148. https://doi.org/10.1016/j.proeng.2014.10.379